CS230: Lecture 4
Xavier initialization
Regularization methods

Kian Katanforoosh
Today’s outline

We will learn how to:

- Prove the initialization methods
- Understand and visualize regularization methods

I. Xavier initialization
II. Regularization methods
III. Announcements
Parameter initialization

Forward propagation (L layers)

We need to initialize the parameters to start the learning process
Parameter initialization

2 important properties of the initialization:

- Break the symmetry
- Not too large/small

Xavier initialization:

\[
W^{[l]} = \text{Normal} \left( \mu = 0, \sigma = \sqrt{\frac{1}{n^{[l-1]}}} \right)
\]

How did they come up with this?
**Parameter initialization**

Goal: Variance stays the same across every layer to prevent the signal from vanishing or exploding

\[ \text{Var}(a^{[l-1]}) = \text{Var}(z^{[l]}) \]
Proof of Xavier initialization

\[ \text{Var}(a^{[l-1]}) = \text{Var}(z^{[l]}) \]

\[ z^{[l]} = W^{[l]} x + b^{[l]} \]
\[ a^{[1]} = g^{[1]}(z^{[1]}) \]
\[ z^{[2]} = W^{[2]} a^{[1]} + b^{[2]} \]
\[ a^{[2]} = g^{[2]}(z^{[2]}) \]
\[ \cdots \]
\[ z^{[L]} = W^{[L]} a^{[L-1]} + b^{[L]} \]
\[ a^{[L]} = g^{[L]}(z^{[L]}) \]

Shape analysis:

\[ z^{[l]} = W^{[l]} a^{[l-1]} = \begin{pmatrix} w_{11}^{[l]} & w_{12}^{[l]} & \cdots & w_{1(n^{[l-1]-1})}^{[l]} & w_{1(n^{[l-1]-1})}^{[l]} & \cdots & w_{1(n^{[l-1]-1})}^{[l]} \\ w_{21}^{[l]} & w_{22}^{[l]} & \cdots & w_{2(n^{[l-1]-1})}^{[l]} & w_{2(n^{[l-1]-1})}^{[l]} & \cdots & w_{2(n^{[l-1]-1})}^{[l]} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ w_{n^{[l]-1}}^{[l]} & w_{n^{[l]-1}1}^{[l]} & \cdots & w_{n^{[l]-1}(n^{[l-1]-1})}^{[l]} & w_{n^{[l]-1}(n^{[l-1]-1})}^{[l]} & \cdots & w_{n^{[l]-1}(n^{[l-1]-1})}^{[l]} \end{pmatrix} \begin{pmatrix} a_1^{[l-1]} \\ a_2^{[l-1]} \\ \vdots \\ a_{n^{[l]-1}}^{[l-1]} \\ a_{n^{[l]-1}}^{[l-1]} \\ \vdots \\ a_{n^{[l]-1}}^{[l-1]} \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^{n^{[l]-1}} w_{1j}^{[l]} a_j^{[l-1]} \\ \sum_{j=1}^{n^{[l]-1}} w_{2j}^{[l]} a_j^{[l-1]} \\ \vdots \\ \sum_{j=1}^{n^{[l]-1}} w_{n^{[l]-1}j}^{[l]} a_j^{[l-1]} \end{pmatrix} \]

\[ z_k^{[l]} = \sum_{j=1}^{n_{[l]-1}} w_{kj}^{[l]} a_j^{[l-1]} \]

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri
Proof of Xavier initialization

$$z_k^{[l]} = \sum_{j=1}^{n^{[l-1]}} w_{kj}^{[l]} a_j^{[l-1]}$$

$$\text{Var}(z_k^{[l]}) = \text{Var}\left(\sum_{j=1}^{n^{[l-1]}} w_{kj}^{[l]} a_j^{[l-1]}\right)$$

$$\text{Var}(XY) = E[X]^2 \text{Var}(Y) + E[Y]^2 \text{Var}(X) + \text{Var}(X)\text{Var}(Y)$$

$$\text{Var}(w_{kj}^{[l]} a_j^{[l-1]}) = E[a_j^{[l-1]}]^2 \text{Var}(w_{kj}^{[l]}) + \text{Var}(w_{kj}^{[l]})^2 \text{Var}(a_j^{[l-1]}) + \text{Var}(w_{kj}^{[l]}).\text{Var}(a_j^{[l-1]})$$

$$\text{Var}(z^{[l]}) = n^{[l-1]} \text{Var}(W^{[l]})\text{Var}(a^{[l-1]})$$

$$\text{Var}(W^{[l]}) = \frac{1}{n^{[l-1]}}$$

$$\text{Var}(w_{kj}^{[l]} a_j^{[l-1]}) = E[a_j^{[l-1]}]^2 \text{Var}(w_{kj}^{[l]}) + \text{Var}(w_{kj}^{[l]})^2 \text{Var}(a_j^{[l-1]}) + \text{Var}(w_{kj}^{[l]}).\text{Var}(a_j^{[l-1]})$$

$$= \left(E[a_j^{[l-1]}]^2 + \text{Var}(a_j^{[l-1]})\right)\text{Var}(w_{kj}^{[l]})$$

$$= E[(a_j^{[l-1]})^2] \text{Var}(w_{kj}^{[l]})$$

$$\text{Var}(z^{[l]}) = n^{[l-1]} \text{Var}(W^{[l]})E[(a_j^{[l-1]})^2]$$

- Weights i.i.d
- Inputs i.i.d
- Weights/Inputs mutually independent

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri
Relation between variance of the input and variance of the output in RELU

\[ \text{Var}(z^{[l]}) = n^{[l-1]} \text{Var}(W^{[l]}) \text{E}[(a^{[l-1]})^2] = \frac{n^{[l-1]}}{2} \text{Var}(W^{[l]}) \text{Var}(a^{[l-1]}) \]

\[ E[g(x)^2] = \int_{x=-\infty}^{x=+\infty} g(x)^2 \, p(x) \, dx = \int_{x=-\infty}^{x=+\infty} \max(0,x)^2 \, p(x) \, dx \]

\[ = \int_{x=0}^{x=+\infty} x^2 \, p(x) \, dx = \frac{1}{2} \int_{x=-\infty}^{x=+\infty} x^2 \, p(x) \, dx \]

\[ = \frac{1}{2} \int_{x=-\infty}^{x=+\infty} (x - \mathbb{E}[x])^2 \, p(x) \, dx \]

\[ = \frac{1}{2} \text{Var}(x) \quad \text{E}[X] = \sum_{i=1}^{n} x_{i} \, P(x_{i} = i) \]
Proof of Xavier initialization

\[ z_k^{[l]} = \sum_{j=1}^{n^{[l-1]}} w_{kj}^{[l]} a_j^{[l-1]} \]

\[ \text{Var}(z_k^{[l]}) = \text{Var} \left( \sum_{j=1}^{n^{[l-1]}} w_{kj}^{[l]} a_j^{[l-1]} \right) = \sum_{j=1}^{n^{[l-1]}} \text{Var}(w_{kj}^{[l]} a_j^{[l-1]}) \]

- Weights i.i.d
- Inputs i.i.d
- Weights/Inputs mutually independent

\[ \text{Var}(w_{kj}^{[l]} a_j^{[l-1]}) = \mathbb{E}[a_j^{[l-1]}]^2 \text{Var}(w_{kj}^{[l]}) + \mathbb{E}[w_{kj}^{[l]}]^2 \text{Var}(a_j^{[l-1]}) + \text{Var}(w_{kj}^{[l]}).\text{Var}(a_j^{[l-1]}) \]

\[ \text{Var}(z^{[l]}) = n^{[l-1]} \text{Var}(W^{[l]}) \text{Var}(a^{[l-1]}) \]

\[ \text{Var}(W^{[l]}) = \frac{1}{n^{[l-1]}} \]

\[ \text{Var}(w_{kj}^{[l]} a_j^{[l-1]}) = \mathbb{E}[a_j^{[l-1]}]^2 \text{Var}(w_{kj}^{[l]}) + \mathbb{E}[w_{kj}^{[l]}]^2 \text{Var}(a_j^{[l-1]}) + \text{Var}(w_{kj}^{[l]}).\text{Var}(a_j^{[l-1]}) \]

\[ \text{Var}(W^{[l]}) = \frac{2}{n^{[l-1]}} \]

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri
Proof of Xavier initialization

This started as a forward propagation analysis

\[
\text{Var}(z^{[L]}) = n^{[L-1]} \text{Var}(W^{[L]}) \text{Var}(a^{[L-1]}) = n^{[L-1]} \text{Var}(W^{[L]}) \text{Var}(z^{[L-1]})
\]

\[
= (n^{[L-1]} \text{Var}(W^{[L]}))(n^{[L-2]} \text{Var}(W^{[L-1]})) \text{Var}(z^{[L-2]})
\]

\[
= ... \]

\[
= \text{Var}(z^{[1]}) \prod_{l=2}^{L} n^{[l-1]} \text{Var}(W^{[l]})
\]

Average:

\[
\text{Var}(W^{[l]}) = \frac{2}{n^{[l]} + n^{[l-1]}}
\]

With a backpropagation analysis:

\[
\text{Var}\left( \frac{\partial J}{\partial a^{[L]}} \right) = \text{Var}\left( \frac{\partial J}{\partial a^{[1]}} \right) \prod_{l=1}^{L-1} n^{[l]} \text{Var}(W^{[l]})
\]

\[
\text{Var}(W^{[l]}) = \frac{1}{n^{[l]}}
\]

Definition Regularization: *technique to help a model generalize better*

<table>
<thead>
<tr>
<th></th>
<th>Training accuracy</th>
<th>Testing accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>98%</td>
<td>60%</td>
</tr>
<tr>
<td>Model regularized</td>
<td>92%</td>
<td>84%</td>
</tr>
<tr>
<td>Model very-well regularized</td>
<td>90%</td>
<td>89%</td>
</tr>
</tbody>
</table>

Techniques: *Dropout, L1, L2, data augmentation, early stopping, …*
### Regularization methods: L1 and L2

**Goal**: keeping the weights small

<table>
<thead>
<tr>
<th>Loss function</th>
<th>Gradient descent update rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ J_{L2}(W, b) = J(W, b) + \frac{1}{2} \lambda W^T W ]</td>
<td>[ W = W - \alpha \frac{\partial J_{L2}}{\partial W} = W - \alpha \left( \frac{\partial J}{\partial W} + \lambda W \right) = W (1 - \lambda \alpha) - \alpha \frac{\partial J}{\partial W} \leq 1 ]</td>
</tr>
</tbody>
</table>

What about L1?
Regularization methods: L1 and L2

Goal: keeping the weights small

Loss function

\[ J_{L1}(W, b) = J(W, b) + \frac{1}{2} \lambda ||W||_1 \]

\[ \frac{\partial}{\partial W} J_{L1}(W, b) = \frac{\partial J(W, b)}{\partial W} + \lambda \text{sign}(W) \]

Gradient descent update rule

\[ W = W - \alpha \frac{\partial J_{L1}}{\partial W} \]

\[ = W - \alpha \left( \frac{\partial J}{\partial W} + \lambda \text{sign}(W) \right) \]

\[ = W - \alpha \frac{\partial J}{\partial W} - \alpha \lambda \text{sign}(W) \]
Regularization methods: L1 and L2

**Goal:** keeping the weights small

**Loss function**

\[
J_{L_1}(W, b) = J(W, b) + \frac{1}{2} \lambda \|W\|_1
\]

\[
\frac{\partial}{\partial W} J_{L_1}(W, b) = \frac{\partial J(W, b)}{\partial W} + ? \; \text{sign}(W)
\]

**Gradient descent update rule**

\[
W = W - \alpha \frac{\partial J_{L_1}}{\partial W}
\]

\[
= W - \alpha \left( \frac{\partial J}{\partial W} + \lambda \text{sign}(W) \right)
\]

\[
= W - \alpha \frac{\partial J}{\partial W} - \alpha \lambda \text{sign}(W)
\]
Visualizing regularization (L1/L2)

Minimum of $J$

Minimum of $J_{L2}$

Minimum of $\lambda W^T W$

Minimum of $J_{L1}$

Minimum of $\lambda \|W\|_1$
Early stopping

How is that a regularizer?

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri
Visualizing early stopping

Minimum of $J$

Best training loss we can do

Starting value of the cost

$\alpha \times (\text{num\_iterations})$
Data augmentation

More data generally means better generalization

Tricky problem 1

• I augmented my data, but my algorithm’s performance is worse than before

Tricky problem 2

• We are a car manufacturing company building a personal vocal assistant, we observe that our model doesn’t generalize well.

Tricky problem 3

• Data synthesis in speech recognition and trigger word detection.
Check out the project example code (cs230-stanford.github.io)

For Thursday 02/08, 9am:

C2M2
• Quiz: Optimization algorithms
• Programming Assignment: Optimizations

C2M3
• Quiz: Hyperparameter tuning, batchnorm, programming frameworks
• Programming Assignment: Tensorflow

Tonight:
• Project proposal
• Fill-in AWS Form to get GPU credits (those who forgot